

# Technologies web et XML

\* par R. LEHN, N. NORMAND et M. PERREIRA DA SILVA

Version PDF des slides

# Organisation des enseignements

## Les cours (1/2)

1. **MPDS** - Introduction + début HTML
2. **MPDS** - HTML + CSS
3. **MPDS** - Fin de CSS + HTTP et Websocket
4. **NN** - Introduction JavaScript
5. **NN** - Fonctions
6. **NN** - Asynchronisme
7. **RL** - Outils, tests unitaires et intégration continue
8. **RL** - XML
9. **RL** - DOM
10. **MPDS** - Programmation côté serveur + Services Web
11. **NN** - POO par prototype
12. **RL** - XSL + XPATH

# Organisation des enseignements

## Mes cours

### 1. Introduction

- Historique, définitions et vue globale des technos web

### 2. Technologies front-end : affichage

- HTML5
- CSS + Media queries
- responsive web design et accessibilité

### 3. Protocoles web

- HTTP(S), MIME, Cookies, WebSocket

### 4. Programmation coté serveur

- Rôle d'un serveur, CGI, PHP, **Nodejs**, etc.

### 5. Services web et protocoles associés

- Architectures par composants, SOAP, REST

# Introduction

# Le web?

- Web ≠ Internet
  - l'Internet = interconnexion de dizaines de milliers de réseaux autonomes
    - Différents protocoles de communication : IP, TCP, DNS, FTP, HTTP, Telnet, etc.
  - le Web = un système hypertexte public fonctionnant sur l'Internet
    - Deux protocoles principaux : HTTP et WebSocket
    - Et d'autres : Ethernet, IP, TCP, UDP, FTP, etc.
- Permet de mettre des ressources à disposition des utilisateurs
  - Texte
  - Images
  - Vidéos
  - etc.

## Quelques dates

- **1967** : Début du programme ARPANET au ministère de la défense des Etats-Unis
- **1979** : Création du réseau civil USENET (newsgroup)
- **1984** : Scission d'ARPANET entre MILNET (militaire) et INTERNET
- **1989** : Invention du Web (système hypertexte) au CERN par Tim Berners-Lee
- **1991** : Apparition des fournisseurs d'accès commerciaux
- **1993** : Premier navigateur graphique (NCSA Mosaic)

# Le web : un écosystème standardisé

## Le W3C

- World Wide Web consortium (<http://www.w3.org/>)
  - Fondé en 1994
  - Président : Tim Berners-Lee (ancien du CERN)
  - 384 membres (dont Microsoft, Mozilla, Apple, Opera, Google, etc.)
- Objectif : créer et promouvoir les standards du web
- Standard W3C = Recommandation
  - Différentes étapes avant recommandation finale (draft, candidate, proposed, etc.)





# Le web : un écosystème standardisé

## Le W3C

- Nombreuses technologies standardisées
  - DOM, (X)HTML, PNG, XML (XPath, XQuery, etc.), SVG, SOAP, WSDL, etc.
- Documents de normalisation en ligne
  - Ex HTML 5: <http://www.w3.org/TR/html5/>
  - Ex CSS 2.1: <http://www.w3.org/TR/CSS2/>
- Outils de validation (gratuits)
  - HTML: <http://validator.w3.org/>
  - CSS: <http://jigsaw.w3.org/css-validator/>
  - Mobile-friendly: <http://validator.w3.org/mobile/>

\* Aller regarder un standard

# Le web : un écosystème standardisé

## Le WaSP

- Web Standards Project (<http://www.webstandards.org/>)
  - Fondé en 1998
- Objectif : respect des standards par les navigateurs
- Les outils du WaSP
  - ACID test (actuellement ACID3)
    - <http://www.acidtests.org/>
  - Évalue : HTML, DOM, CSS, SVG, ECMAScript (JavaScript)



# Le web : un écosystème standardisé

## L'IETF et les RFC

- Internet Engineering Task Force
  - Groupe informel
  - Création de la plupart de RFC (Request For Comment)
- Certaines RFC sont des standards (liste complète dans [RFC 5000](#))
  - UTF-8: [RFC 3629](#)
  - MIME: [RFC 2045](#)
  - HTTP: [RFC 2616](#)
  - etc.





please don't hurt the web  
**use open standards**

mozilla developer center  
[developer.mozilla.org](https://developer.mozilla.org)



mozilla  
**Firefox**

# Terminologie et définitions

- **Hôte** : Ordinateur relié au réseau
- **Client** : Ordinateur envoyant des requêtes auxquelles un autre ordinateur va répondre
- **Serveur** : Ordinateur qui écoute et traite les requêtes du client
- **Navigateur** : programme informatique permettant de consulter des ressources sur le Web
- **Ressource** : "Objet" qui peut être localisé et atteint à travers le réseau
- **Nom de domaine** : Identifie un ensemble d'ordinateurs reliés à Internet et possédant une caractéristique commune

# Terminologie et définitions (suite)

- **Hypertexte** : Système contenant des nœuds reliées entre eux par des liens
- **Hyperlien** : Lien qui relie deux nœuds entre eux
- **HTTP** (Hypertext Transfer Protocol) : Protocole de communication client-serveur principalement utilisé sur le web
- **XML** (eXtensible Markup Language) : Langage de balisage générique.
- **HTML** (Hypertext Markup Language) : Langage de balisage conçu pour représenter des pages web. Permet de définir de l'Hypertexte.
- **CSS** (Cascading Style Sheets) : Langage de description permettant de mettre en forme les documents HTML et CSS
- **SVG** (Scalable Vector Graphics): Format de données basé sur XML, permettant de décrire des images vectorielles
- **PNG** (Portable Network Graphics) : Format d'images matricielles permettant une compression sans pertes

# URI, URL et URN

- **URI** (Uniform Resource Identifier) : Identifie une ressource (physique ou abstraite) sur un réseau (**RFC 3986**)
- **Schéma** : Type de ressource identifiée par l'URI (ex: http, ftp, mailto, about). Ne référence pas forcément un protocole
- **URL** (Uniform Resource Locator) : Est une URI, mais fournit en plus un moyen d'agir sur la ressource ou d'en obtenir une représentation. Décrit également le mode d'accès à cette ressource
  - Ex1: <http://web.polytech.univ-nantes.fr/>
  - Ex2: mailto:matthieu.perreiradasilva@univ-nantes.fr
- **URN** (Uniform Resource Name) : Est une URI qui identifie une ressource par son nom dans un espace de nom
  - Ex: isbn:978-2-7540-3849-2

# URL et protocole HTTP

- **Absolue** : schéma://utilisateur:motdepasse@domaine:port/chemin?  
requête#fragment
  - Ex: `http://joe:bar@www.univ-nantes.fr:80/polytech/dpts/info?enseignant=perreiradasilva-m&cours=technos-web#slide4`
- **Relative** : chemin?requête#fragment
  - Ex1: `ici`
  - Ex2: `/ici`
  - Ex3: `./ici?query=something`
  - Ex4: `../la/fichier.html`



# Encodage des caractères

- Représentation numérique des caractères
  - Ex: 'A' = 0x41 en ASCII (hexadécimal)
- **ASCII**: caractères de base (US) codés sur 7bits
- **Variantes de l'ASCII** sur 8 bits
  - Nécessité de préciser le **jeu de caractères** (character set / codepage)
  - Ex: ISO 8859-1 (europe occidentale) et 8859-15 (variation avec €)
- **UTF-8**: encodage universel à taille variable (RFC 3629)
  - De 8 à 32 bits, rétrocompatible avec ASCII
    - Ex: 'A' = 0x41, 'à' = 0xC3 0xA0, '€' = 0xE2 0x82 0xAC

# Percent-encoding

- Les URI ne sont codées qu'à partir d'un jeu limité de caractères ASCII
  - Caractères réservés : ont une signification particulière

! # \$ % & ' ( ) \* + , / : ; = ? @ [ ]

- Caractères non réservés : pas de signification particulière

A B C D E F G H I J K L M N O P Q R S T  
U V W X Y Z

a b c d e f g h i j k l m n o p q r s t  
u v w x y z

0 1 2 3 4 5 6 7 8 9 - \_ . ~

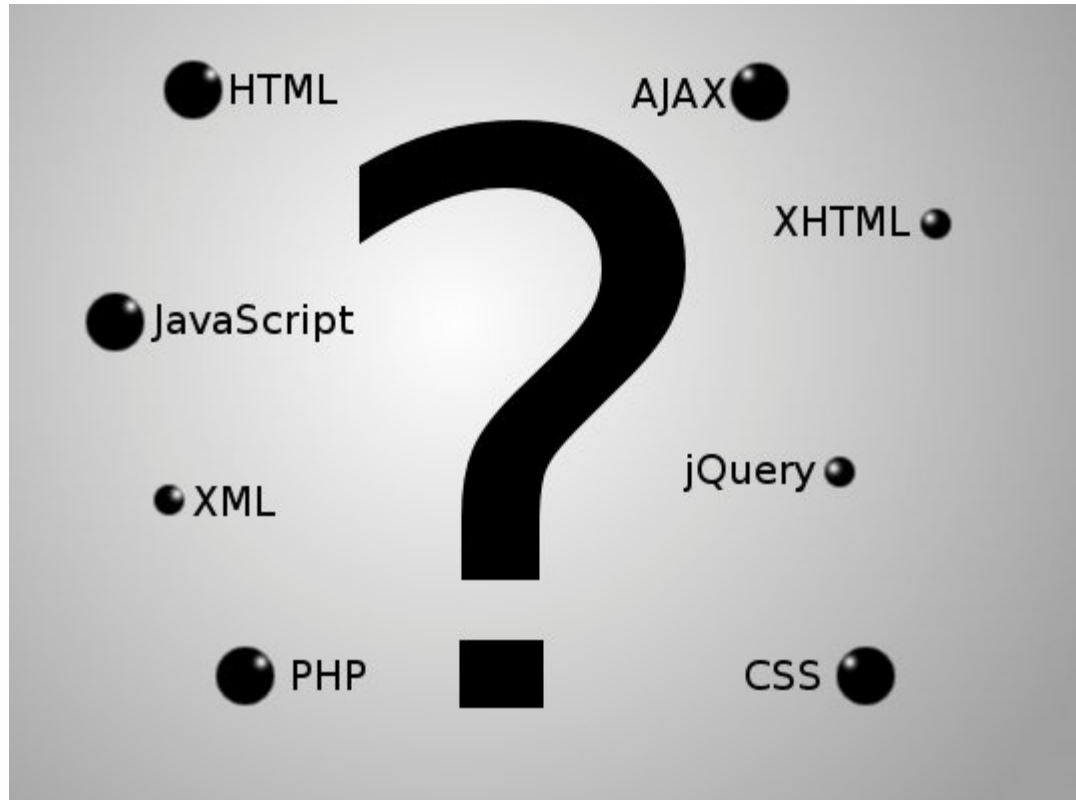
# Percent-encoding (suite)

- Pour tous les autres caractères ou pour utiliser les caractères réservés sans qu'ils ne soient interprétés on utilise le percent encoding
  - Règle générale: %code\_UTF-8 (en hexadécimal)
    - Ex1: espace = %20
    - Ex2: % = %25
    - Ex3: € = %E2%82%AC
  - Même chose pour les caractères réservés

!	#	\$	&	'	(	)	*	+
%21	%23	%24	%26	%27	%28	%29	%2A	%2B

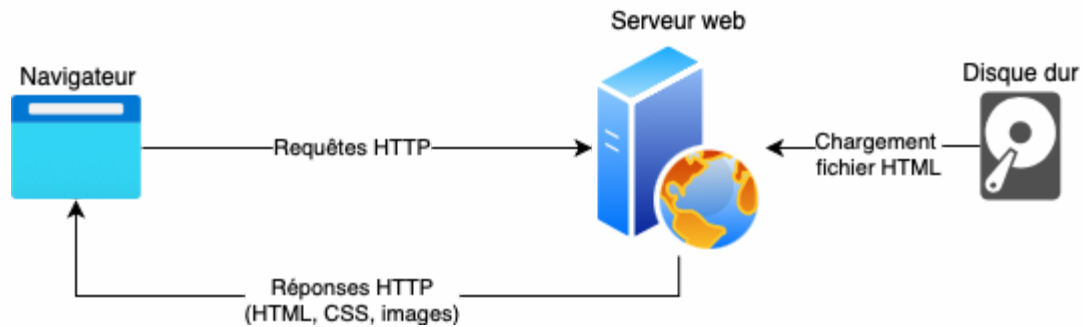
,	/	:	;	=	?	@	[	]
%2C	%2F	%3A	%3B	%3D	%3F	%40	%5B	%5D

# Programmer pour le web ?



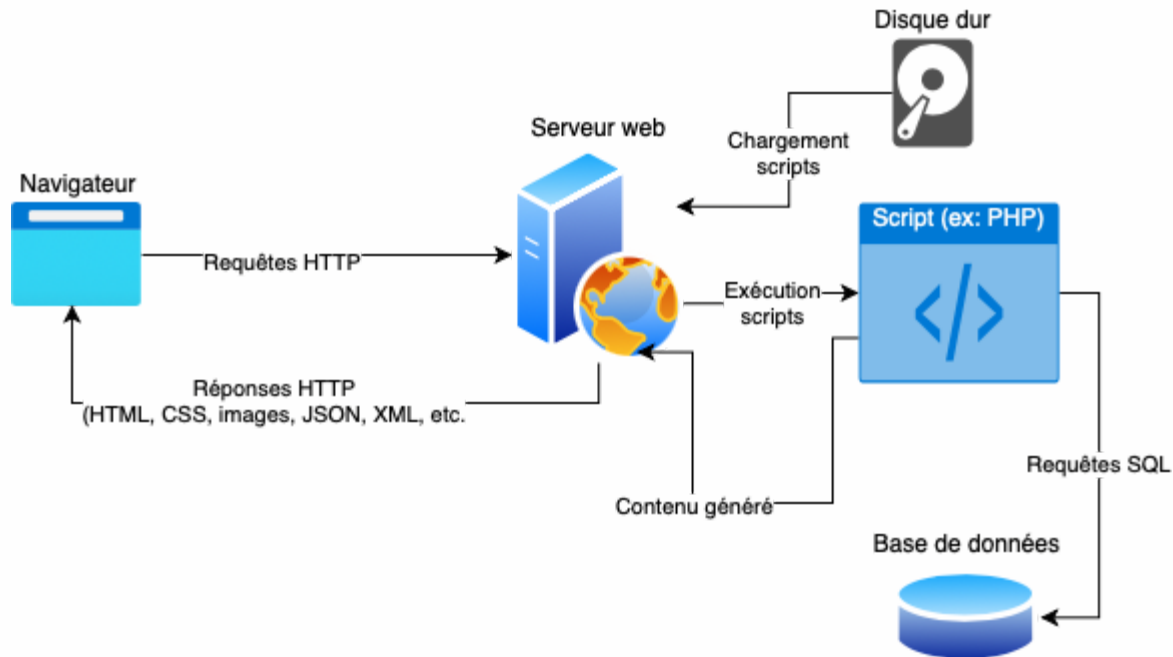
# Site statique

- Pages HTML (+CSS) créées à l'avance
- Le serveur web ne fait qu'envoyer le contenu des fichiers
- Pas de possibilité d'adaptation aux requêtes du client
- Le web des années 90...



# Site dynamique

- Pages HTML générées à la demande par le serveur
- Exécution de scripts sur le serveur (PHP, ASP, Python, etc.)
- Possibilité d'adapter le contenu à la requête envoyée par le client
- Interaction du serveur web avec une base de données
- Le web des années 2000+.

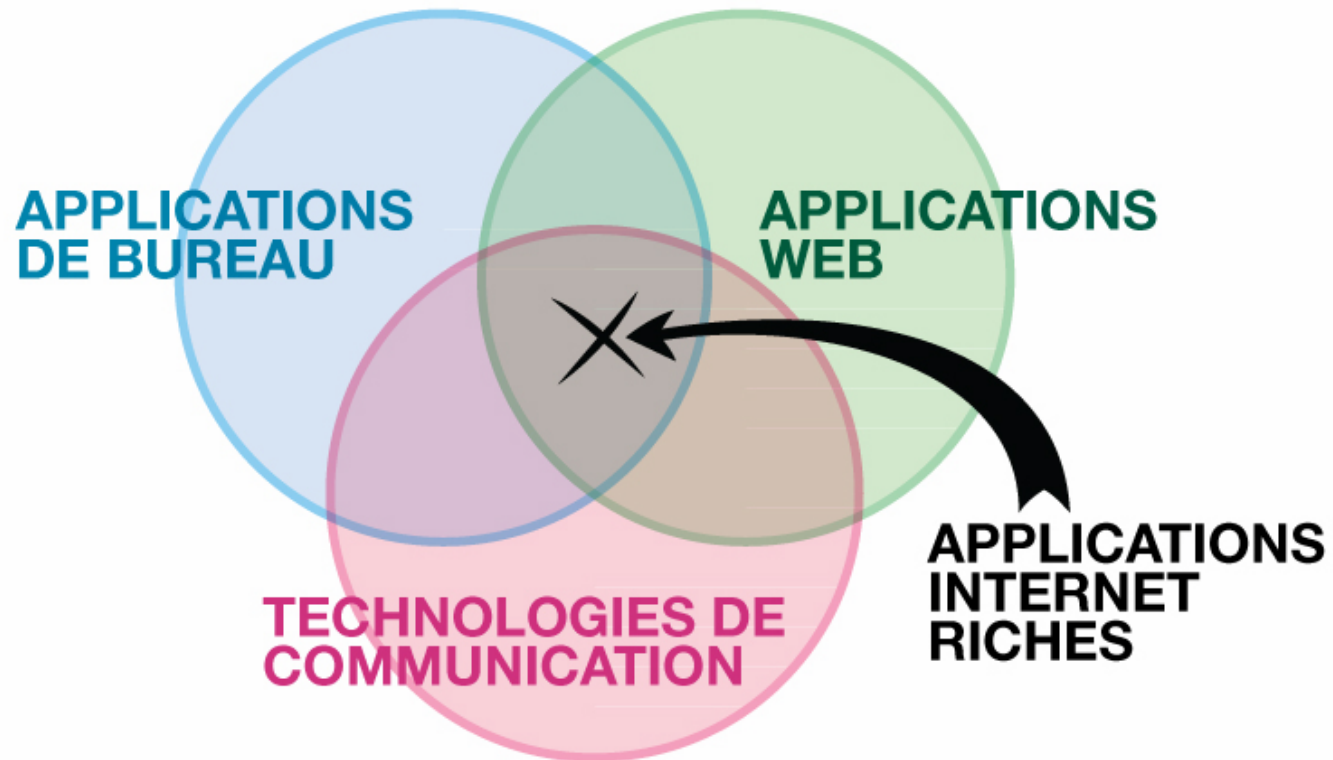


# Deux façons de scripter

## Client ou serveur ?

- Historiquement : scripts exécutés sur le serveur (Server Side Scripting)
  - PHP, ASP, J(2)EE, etc.
- Plus récemment : DOM + évolutions de JavaScript
  - Nombreuses interactions coté client (Client Side Scripting)
  - Utilisation de requêtes pour récupérer des données (ex: AJAX)
    - Toujours besoin d'un serveur web pour les données / l'API
  - Applications internet riches (RIA)
    - Ex: Gmail

# Applications internet riches (RIA)





# RIA pour les nuls ?

## Applets et plugins (obsolète)

- JavaFX (Oracle)
  - Basé sur Java
- Flex (Adobe)
  - Basé sur Flash
- Silverlight (Microsoft)
  - Basé sur le Framework .Net



Microsoft®  
Silverlight™

# Applets et plugins (obsolète)

## Avantages

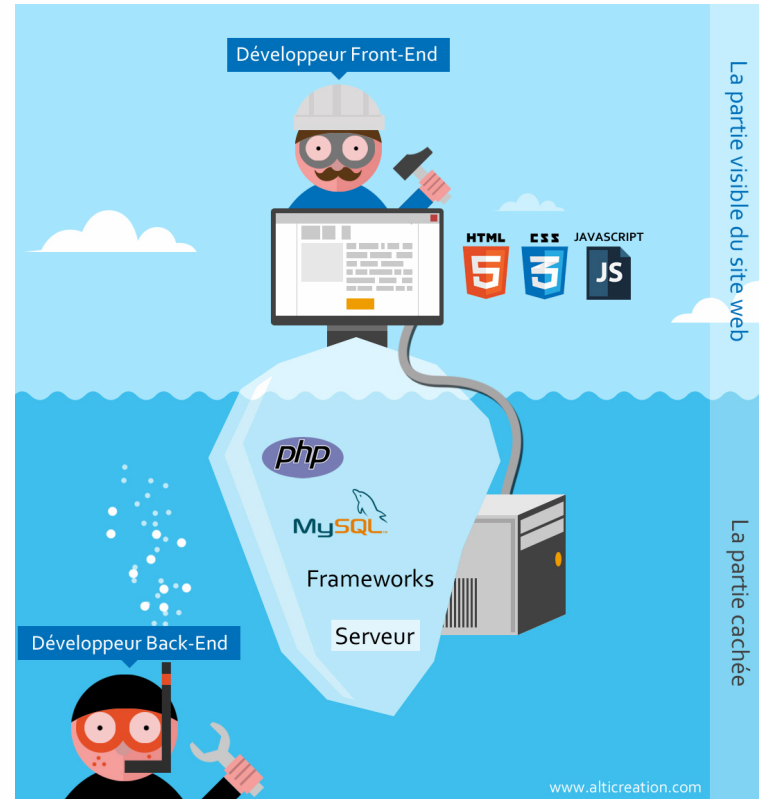
- Richesse graphique
  - Animations, etc.
- UX proche d'une application "Desktop"
  - Interface, réactivité, etc.
- Ajout de fonctionnalités non / mal supportées par les navigateurs
  - Ex: streaming vidéo

## Inconvénients

- Nécessité de télécharger un plugin
- Temps de téléchargement initial de l'application
- Risques de failles de sécurité accrus
- Accessibilité et référencement

# Les deux faces du développement web...

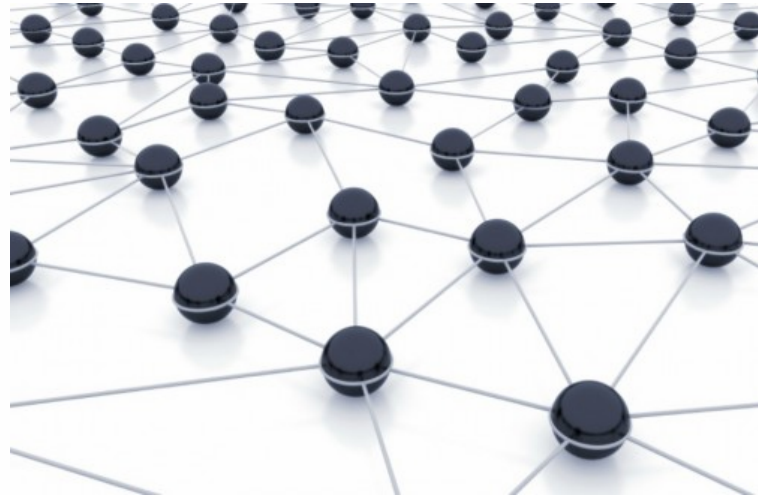
- Frontend
  - (Très) proche du design
  - Integration
  - Interactions
  - HTML / CSS / Javascript
- Backend
  - Le "Vrai" développeur web...
  - Accès aux données
  - Code métier
  - PHP / MySQL, J2EE, NodeJS, Python



# Les technologies web c'est aussi...

## Les systèmes distribués

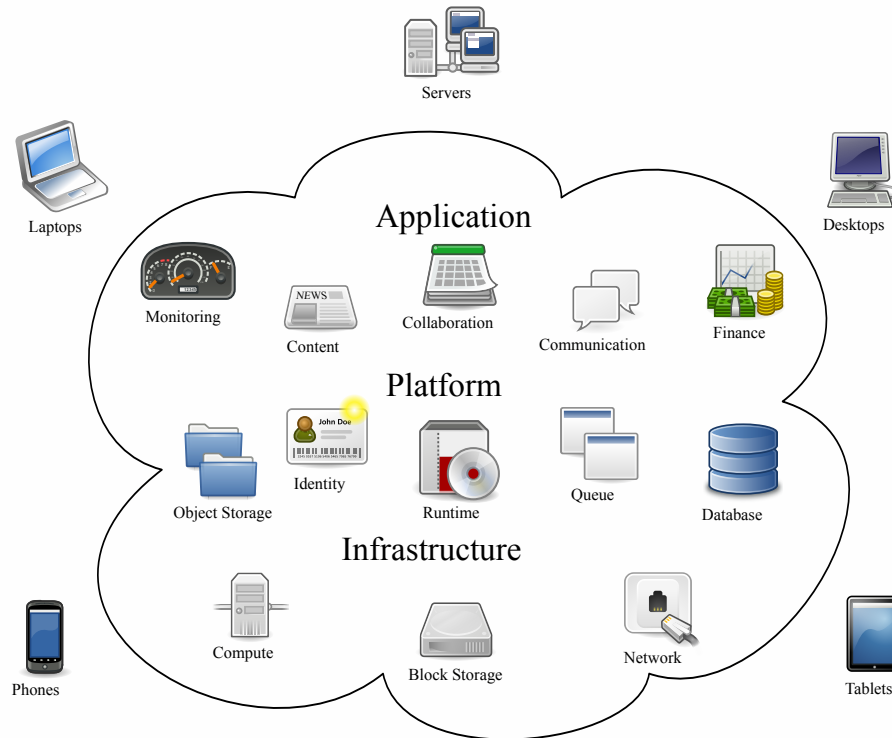
- Pas que des sites web...
  - Applications mobiles
  - Clients lourds
  - Services / API web
  - Cloud computing
  - Etc.
- Technologies réutilisées
  - HTTP, XML, JSON, MIME, URI, etc.



# Cloud computing

## Dématérialisation de l'informatique

- Bonus, pour votre culture:
  - <https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/>



# Pour votre culture

- Faits marquants du web
  - <http://www.webdirections.org/history/>
  - <http://www.pewinternet.org/2014/03/11/world-wide-web-timeline/>



*Ex: (1996) Dancing baby : première vidéo virale*

# This is (almost) the end...

Next : HTML + CSS

